Assignment 6: Unsupervised Clustering

UVA CS 4501 :
Machine Learning (Fall 2018)

Out: Nov. 25 2018
Due: Dec 12th, Sat midnight 11:59pm, 2018 @ Collab

a *The assignment should be submitted in the PDF format through Collab. If you prefer hand-writing QA parts of answers, please convert them (e.g., by scanning or using PhoneApps like officeLens) into PDF form.*

b *For questions and clarifications, please post on piazza.*

c *Policy on collaboration:*

Homework should be done individually: each student must hand in their own answers. It is acceptable, however, for students to collaborate in figuring out answers and helping each other solve the problems. We will be assuming that, with the honor code, you will be taking the responsibility to make sure you personally understand the solution to any work arising from such collaboration.

d *Policy on late homework: Homework is worth full credit at the midnight on the due date. Each student has extension days to be used at his or her own discretion throughout the entire course. Your grades would be discounted by 15% per day when you use these late days.*

1 Unsupervised Learning with Clustering

In this programming assignment, you are required to implement clustering algorithm: K-means Clustering. A ZIP file has been provided ("data_sets_clustering.zip") that includes two different datasets. Please follow all instructions for submitting the source code.

You are required to submit a source-code file "clustering.py" containing the necessary functions for training and evaluations. The maximum number of iterations to be performed for both algorithms is 1000.

DO NOT use scikit-learn package in this problem and please implement yours from scratch.

1.1 Data description

We have provided two different datasets for evaluating your clustering code.

- **Dataset 1** : The first dataset consists of height and weight data for average people and baseball players. First column contains human height (inches) and second column has human weight (lbs). The third column has true labels of samples and will be used only for evaluating the clusters you discover.

1.2 load data

- (Q1) You are required to code the following function for loading datasets:
  X = loadData(fileDj)

1.3 K-means Clustering

- (Q2) Next, code the following function to implement k-means clustering:
  labels = kmeans(X, k, maxIter)
  Here X is the input data matrix, k is the number of clusters and maxIter is the maximum number of the iterations selected by you (max value =1000).

- (Q3) Implement k-means clustering for **Dataset 1**(use first two columns in the file as input) and use scatter() function in the matplotlib package to visualize the result. The two clusters must be in different colors.

- (Q4) Implement k knee-finding (also called elbow-finding) method for **Dataset 1** and k = {1,2,...,6} to select value of k (number of clusters) and plot graph for k versus objective function value (e.g. Slide 99, Lecture 20).

- (Q5) Now, code the following function to calculate the purity metric for the evaluation of results:
  puritymetric = purity(labels, trueLabels)
  Use this function to evaluate the results of (Q3)

1.4 How will your code be checked?

We will run the following command: "python clustering.py DatasetDirectoryFullPath" and your code should print the following results

- the scatter plots from (Q3)
- k knee-finding plot in (Q4)
- ALL purityMetric values for results obtained in (Q3)

1.5 Submission

Please submit your source code as "clustering.py" and PDF report containing your written answers via collab. In the report, you should include the following contents

- ALL scatter plots generated in (Q3)
- k knee-finding plot in (Q4)
- ALL purityMetric values for results obtained in (Q3)

We are posting the solution of following QAs to help you prepare for the final exam. You are still required to copy the answers and please provide your own understanding of the answers in your submission.

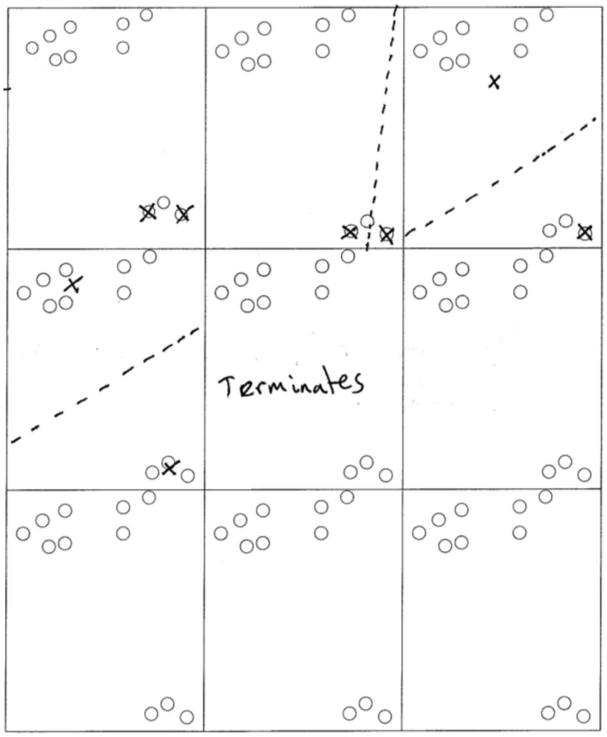
2 Sample Exam Questions:

Each assignment covers a few sample exam questions to help you prepare for the midterm and the final. (Please do not bother by the information of points in some the exam questions.)

Question: 1. K-means and Gaussian Mixture Models

(a) Run k-means manually for the following dataset, where $k = 2$. Circles are data points and squares are the initial cluster centers. Draw the cluster centers and the decision boundaries that define each cluster. Use as many pictures as you need until convergence.
Note: Execute the algorithm such that if a mean has no points assigned to it, it stays where it is for that iteration.

Answer:

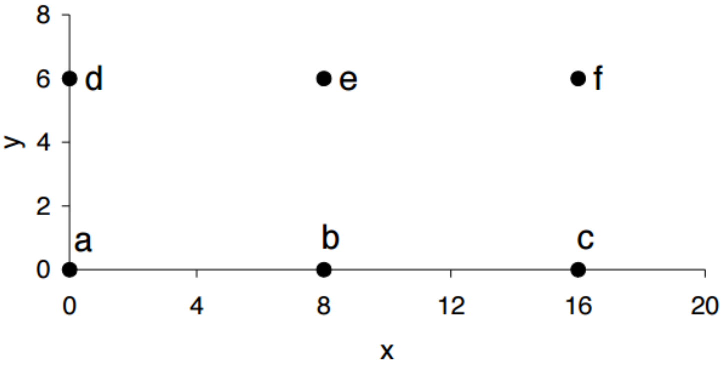


Question: 2. K-means Clustering

There is a set S consisting of 6 points in the plane shown below where $a = (0, 0), b = (8, 0), c = (16, 0), d = (0, 6), e = (8, 6), f = (16, 6)$. Now we run the k-means algorithm on these points with $k = 3$. The algorithm uses the Euclidian distance metric (i.e. the straight line distance between two points) to assign each point to its nearest centroid. Ties are broken in favor of the centroid to the left/down. We define the following two definitions:

- A **k-starting configuration** is a subset of k staring points from S that form the initial centroids, e.g. $\{a, b, c\}$.
- A **k-partition** is a partition of S into k non-empty subsets, e.g. $\{a, b, e\}$, $\{c, d\}$, $\{f\}$ is a 3-partition.

Clearly any k-partition induces a set of k centroids in the natural manner. A k-partition is called *stable* if a repetition of the k-means iteration with the induced centroid leaves it unchanged.



(a) How many 3-starting configurations are there? (Remember, a 3-starting configuration is just a size 3 subset of the 6 datapoints.

Answer: $C_6^3 = 20$

(b) Fill in the following table:

Answer:

3-partition	Is it stable?	An example 3-starting configuration that can arrive at the 3-partition after 0 or more iterations of k-means (or write "none" if no such 3-starting configuration)	The number of unique starting configurations that can arrive at the 3-partition.
$\{a,b,e\}$,$\{c,d\}$,$\{f\}$	N	none	0
$\{a,b\}$,$\{d,e\}$,$\{c,f\}$	Y	{b,c,e}	4
$\{a,d\}$,$\{b,e\}$,$\{c,f\}$	Y	{a,b,c}	8
$\{a\}$,$\{d\}$,$\{b,c,e,f\}$	Y	{a,b,d}	2
$\{a,b\}$,$\{d\}$,$\{c,e,f\}$	Y	none	0
$\{a,b,d\}$,$\{c\}$,$\{e,f\}$	Y	{a,c,f}	1

Question: 3. Decision Trees The following dataset will be used to learn a decision tree for predicting whether a person is happy (H) or sad (S) based on the color of their shoes, whether they wear a wig and the number of ears they have.

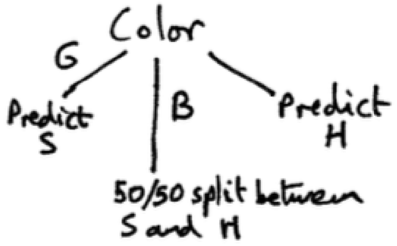
Color	Wig	Num. Ears	(Output) Emotion
G	Y	2	S
G	N	2	S
G	N	2	S
B	N	2	S
B	N	2	H
R	N	2	H
R	N	2	H
R	N	2	H
R	Y	3	H

(a) [2 points] What is $H(Emotion|Wig = Y)$ (where H is entropy)?
Answer: Answer: 1

(b) [2 points] What is $H(Emotion|Ears = 3)$?
Answer: Answer: 0

(c) [3 points] Which attribute would the decision-tree building algorithm choose to use for the root of the tree (assume no pruning)?
Answer: Answer: Color

(d) [3 points] Draw the full decision tree that would be learned from this data (assume no pruning).
Answer: Answer: Color is root node, predict sad if green, happy if red, and 50/50 split if blue.



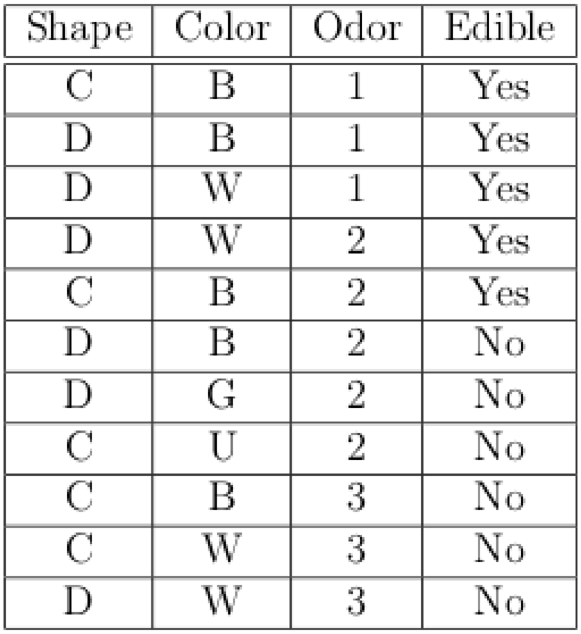
The next two parts do not use the previous example, but are still about decision tree classifiers.

(e) [3 points] Assuming that the output attribute can take two values (i.e. has arity 2) what is the maximum training set error (expressed as a percentage) that any dataset could possibly have?
Answer: Answer: 50%

(f) [3 points] Construct an example dataset that achieves this maximum percentage training set error (it must have two or fewer inputs and five or fewer records).
Answer: Answer: $x : \{0, 0, 1, 1\}$, $y : \{0, 1, 0, 1\}$

Question: 4. Decision Trees

The following dataset will be used to learn a decision tree for predicting whether a mushroom is edible or not based on its shape, color, and odor.



Shape	Color	Odor	Edible
C	B	1	Yes
D	B	1	Yes
D	W	1	Yes
D	W	2	Yes
C	B	2	Yes
D	B	2	No
D	G	2	No
C	U	2	No
C	B	3	No
C	W	3	No
D	W	3	No

(a) [4 points] What is the entropy $H(Edible|Odor = 1 \text{ } or \text{ } Odor = 3)$?
Answer: Answer: 1

(b) [4 points] Which attribute would the ID3 algorithm choose to use for the root of the tree (no pruning)?
Answer: Answer: Odor

(c) [4 points] Draw the full decision tree that would be learned for this data (no pruning).

```
       Odor
       / | \
     1/ 2|  \3
     /   |   \
  Yes  Color  No
      / | | \
     /  | |  \
  W / B|  \G \U
   /   /    \  \
Yes  Shape  No  No
      / \
    C/   \D
    /     \
  Yes     No
```

Answer: see figure:

Question: 5. Decision Trees and Hierarchical Clustering

Assume we are trying to learn a decision tree. Our input data consists of N samples, each with k attributes ($N \gg k$). We define the depth of a tree as the maximum number of nodes between the root and any of the leaf nodes (including the leaf, not the root).

(a) (2 points) If all attributes are binary, what is the maximal number of leaf (decision) nods that we can have in a decision tree for this data. What is the maximal possible depth of a decision tree for this data?

Answer: 2^k. Each feature can only be used once in each path from root to leaf. The maximum depth is O(k).

(b) (2 points) If all attributes are continuous, what is the maximum number of leaf nodes that we can have in a decision tree for this data? What is the maximal possible depth for a decision tree for this data?

Answer: Continuous values can be used multiple times, so the maximum number of leaf nodes can be the same as the number of samples. The maximal depth is N (or N-1 if we do not count root node). In binary trees, the number of leaves is always one more than the number of internal nodes. e.g. a very unbalanced tree, with each split getting a decision child node and an internal node.

(c) (2 points) When using **single link** what is the maximal possible depth of a hierarchical clustering tree for the data in (a). What is the maximal possible depth of such a hierarchical clustering tree for the data in (b).

Answer: When using single link with binary data, we can obtain cases where we are always growing the cluster by 1 node at a time leading to a tree of depth N (N-1 if we do not count root node). This is also clearly the case for continuous values.

(d) (2 points) Would your answers to (c) change if we were using **complete link** instead of **single link**? If so, would it change for both types of data? Briefly explain.

Answer: While the answer for continuous values remain the same (its easy to design a dataset where each new sample is farther from any of the previous samples) for binary data, if k is small compared to N we will not be able to continue to add one node at a time to the initial cluster and so the depth will change to be lower than N. For binary features, when k is small, the number of possible unique samples is 2^k, which is likely to be smaller than N. This means many samples might be overlap with each other.